Amendments to the Claims

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (currently amended) A method for restoring persistently stored objects of an object-oriented environment established in a computer system having a volatile memory and a persistent storage, the method comprising the steps of:

retrieving from ~~said~~ the persistent storage a first list comprising first references to segments, stored in ~~said~~ the persistent storage;

retrieving all segments referenced by ~~said~~ the first references and storing ~~them~~ the segments in ~~said~~ the volatile memory;

saving in ~~said~~ the first list the difference between an old memory address, at which the segment used to reside in the volatile memory, and a new memory address at which ~~said~~ the segment is stored;

retrieving from ~~said~~ the persistent storage a second list comprising second references to blocks, ~~whereby~~ wherein ~~said~~ the blocks contain an object description;

determining which segment contains ~~said~~ the block referenced by a particular element of ~~said~~ the second list;

creating an object in ~~said~~ the volatile memory using ~~said~~ the object description from ~~said~~ the segment and saving a new address of ~~said~~ the created object in ~~said~~ the second list in volatile memory;

initializing ~~said~~ the new object with values taken from ~~said~~ the object description; and

determining ~~said~~ the new addresses of ~~said~~ the new object referenced by the newly-created object and

setting ~~said~~ the new address as the reference in ~~said~~ the new object.

2. (currently amended) The method according to claim 1, ~~whereby~~ ~~said~~ wherein either the first list ~~and/or said~~ or second list ~~are~~ is an ordered ~~lists~~ list or both the second and first lists are ordered lists.

3. (currently amended) The method according to claim 1 or 2, ~~whereby said~~ wherein the first list and~~/or~~ ~~said~~ the second list are organized as a B-tree or either said first list or said second list is an ordered list.

4. (currently amended) The method according to claim 1, ~~whereby~~ wherein the elements of ~~said~~ the first ordered list are indexed by ~~said~~ the first references.

5. (currently amended) The method according to ~~Claim~~ claim 1, whereby each of ~~said~~ the first references corresponds to the old memory address at which the respective segment used to reside in the volatile memory.

6. (currently amended) The method according to ~~Claim~~ claim 1, ~~whereby~~ wherein the elements of ~~said~~ the second ordered list are indexed by ~~said~~ the second references.

7. (currently amended) The method according to ~~one~~ ~~Claim~~ claim 1, ~~whereby~~ wherein each of ~~said~~ the second references corresponds to the old memory address at which the respective block used to reside in ~~said~~ the volatile memory.

8. (currently amended) The method according to ~~Claim~~ claim 1, ~~whereby~~ wherein ~~said~~ the object description is formed by a collection of values owned by an object for the variables belonging to its class.

9. (currently amended) The method according to ~~Claim~~ claim 1, ~~whereby~~ wherein for each value in ~~said~~ the object description of variables having a variable length the method further comprising the steps of:

allocating a number of blocks that allows to keep the actual value of the variable having variable length;

creating a linked list of ~~said~~ the number of blocks;

saving ~~said~~ the value into ~~said~~ the number of blocks; and

storing a reference to the head of the linked list in ~~said~~ the object description.

10. (currently amended) The method according to claim 1, ~~whereby~~ wherein determining the segment that contains ~~said~~ the block referenced by a particular element of ~~said~~ the second list comprises ~~the~~ a step of searching in ~~said~~ the first ordered list ~~(segment map)~~ for the segment that contains ~~said~~ the portion of ~~said~~ the segment ~~(block)~~ referenced by ~~said~~ the element.

11. (currently amended) The method according to claim 1, ~~whereby~~ wherein determining ~~the~~ which segment ~~that~~ contains ~~said~~ the block referenced by a particular element of ~~said~~ the second list further comprises ~~the~~ a step of calculating the new address by adding the reference to ~~said~~ the block, ~~(~~that corresponds to the old memory address~~)~~, and ~~said~~ the difference between ~~said~~ the old memory address and ~~said~~ the new memory address.

12. (currently amended) The method according to claim 1 ~~claims~~ further comprising, ~~whereby determining the new addresses of objects referenced by the newly created object comprises~~ ~~the~~ a step of searching in ~~said~~ the second list ~~(object map)~~ for the element ~~said~~ that contains the new address of the referenced object, that is referenced by the old address of the respective object description.

13. (currently amended) The method according to claim 1, ~~whereby~~ wherein ~~for~~ all references to heads of linked lists, the method further ~~comprising~~ comprises the steps of:

reading all blocks of ~~said~~ the linked list;

allocating memory to store the value of the variable retrieved from the linked list; and storing the value in ~~said~~ the allocated memory.

14. (currently amended) A method for persistently storing objects of an object-oriented environment established on a computer system having a volatile memory and a persistent storage, the method comprising steps of:

allocating in ~~said~~ the volatile memory, segments;

creating a first list comprising first references to ~~said~~ the segments;

creating a second list comprising second references to blocks, wherein the blocks are portions of ~~said~~ the segments;

allocating a block of one of ~~said~~ the segments,

creating an object description for an object by saving values owned by the object of the variables belonging to its class into ~~said~~ the allocated block;

adding a new element to ~~said~~ the second list containing the particular reference to ~~said~~ the object description;

determining the address of another object description of another object referenced in ~~said~~ the object;

setting the address of ~~said~~ the respective object description as the reference in the created object description;

storing ~~said~~ the second list on ~~said~~ the persistent storage;

storing the segments referenced by ~~said~~ the first list on ~~said~~ the persistent storage; and

storing ~~said~~ the first list on ~~said~~ the persistent storage.

15. (currently amended) The method according to claim 14, ~~whereby said~~ wherein the first list ~~and/or said second list are ordered lists~~ and at least one of the first and second lists is an ordered list.

16. (currently amended) The method according to claim 14, ~~whereby said~~ wherein at least one of the first list ~~and/or said~~ and second list ~~are~~ is organized as a B-tree.

17. (currently amended) The method according to claim 14, ~~whereby~~ wherein the elements of ~~said~~ the first ordered list are indexed by ~~said~~ the first references.

18. (currently amended) The method according to claim 14, ~~whereby~~ wherein each of ~~said~~ the first references corresponds to the current memory address at which the respective segment resides in the volatile memory.

19. (currently amended) The method according to claim 14, ~~whereby~~ wherein the elements of ~~said~~ the second ordered list are indexed by ~~said~~ the second references.

20. (currently amended) The method according to claim 14, ~~whereby~~ wherein each of ~~said~~ the second references corresponds to the current memory address at which the respective block resides in ~~said~~ the volatile memory.

21. (currently amended) The method according to claim 14, ~~whereby~~ wherein determining the address of the object description of another object referenced in ~~said~~ the object comprises ~~the~~ a step of searching in ~~said~~ the second ordered list ~~(object map)~~ for the element ~~said~~ that contains the address of the object description of the referenced other object.

22. (currently amended) The method according to claim 14, ~~whereby determining the address of the object description of another object referenced in said object comprises~~ further comprising a step of using a reference to the respective object description provided by each object.

23. (currently amended) The method according to claim 14, ~~whereby~~ wherein for each value of variables comprises a variable length the method further comprises steps of:

allocating a number of portions of one of ~~said~~ the pieces of memory that allows ~~to keep~~ keeping the actual value of the variable length variable;

creating a linked list of ~~said~~ the number of portions;

saving value into ~~said~~ the number of portions; and

storing a reference to the head of the linked list in ~~said~~ the object description.

24. (currently amended) A computer program product stored on a computer usable medium, comprising computer readable program instructions for:

allocating in ~~said~~ the volatile memory segments;

creating a first list comprising first references to ~~said~~ the segments;

creating a second list comprising second references to blocks;

allocating a block of one of ~~said~~ the segments,

creating an object description by saving values owned by the object of the variables belonging to its class into ~~said~~ the allocated block;

adding a new element to ~~said~~ the second list containing the particular reference to ~~said~~ the created object description;

determining the address of the object description of another object referenced in ~~said~~ the object;

setting the address of ~~said~~ the respective object description as the reference in the created object description;

storing ~~said~~ the second list on ~~said~~ the persistent storage;

storing the segments referenced by ~~said~~ the first list on ~~said~~ the persistent storage; and

storing ~~said~~ the first list on ~~said~~ the persistent storage.

25. (new) A method for restoring persistently stored objects of an object-oriented environment established in a computer system having a volatile memory and a persistent storage, the method comprising steps of:

retrieving from the persistent storage a first ordered list comprising first references to segments, stored in the persistent storage, wherein the elements of the first ordered list are indexed by the first references, the elements of the second ordered list are indexed by the second references and each of the first references corresponds to the old memory address at which the respective segment used to reside in the volatile memory;

retrieving all segments referenced by the first references and storing the segments retrieved in the volatile memory;

saving in the first list the difference between an old memory address, at which the segment used to reside in the volatile memory, and a new memory address at which the segment is stored;

retrieving from the persistent storage a second ordered list comprising second references to blocks, wherein the blocks contain an object description;

determining which segment contains the block referenced by a particular element of the second list;

creating an object in the volatile memory using the object description from the segment and saving a new address of the created object in the second list in volatile memory;

initializing the new object with values taken from the object description;

determining the new addresses of the new object referenced by the newly- created object;

setting the new address as the reference in said new object.